# Machine Learning Foundation

## Car Evaluation

Name – Gagan Raturi

Reg. No. – 11808991

Roll No. – 57

Section – KM002

Email – eshuraturi@gmail.com

Submitted to – **Professor Sanjay Kumar Singh**

# Data Set Description

## Car Evaluation Data Set

*Download*: Data Folder, Data Set Description

**Abstract**: Derived from simple hierarchical decision model, this database may be useful for testing constructive induction and structure discovery methods.

| | | | | | |
|---|---|---|---|---|---|
| **Data Set Characteristics:** | Multivariate | **Number of Instances:** | 1728 | **Area:** | N/A |
| **Attribute Characteristics:** | Categorical | **Number of Attributes:** | 6 | **Date Donated** | 1997-06-01 |
| **Associated Tasks:** | Classification | **Missing Values?** | No | **Number of Web Hits:** | 1359879 |

1. Title: Car Evaluation Database

2. Sources:

(a) Creator: Marko Bohanec
(b) Donors: Marko Bohanec (marko.bohanec@ijs.si)
Blaz Zupan (blaz.zupan@ijs.si)
(c) Date: June, 1997

3. Relevant Information Paragraph:

Car Evaluation Database was derived from a simple hierarchical decision model originally developed for the demonstration of DEX (M. Bohanec, V. Rajkovic: Expert system for decision making. Sistemica 1(1), pp. 145-157, 1990.). The model evaluates cars according to the following concept structure:

CAR car acceptability

. PRICE overall price

. . buying buying price

. . maint price of the maintenance

. TECH technical characteristics

. . COMFORT comfort

. . . doors number of doors

. . . persons capacity in terms of persons to carry

. . . lug_boot the size of luggage boot

. . safety estimated safety of the car

The Car Evaluation Database contains examples with the structural information removed, i.e., directly relates CAR to the six input attributes: buying, maint, doors, persons, lug_boot, safety.

Because of known underlying concept structure, this database may be particularly useful for testing constructive induction and structure discovery methods.

4. Number of Instances: 1728

(instances completely cover the attribute space)

5. Number of Attributes: 6

6. Attribute Values:

buying v-high, high, med, low

maint v-high, high, med, low

doors 2, 3, 4, 5-more

persons 2, 4, more

lug_boot small, med, big

safety low, med, high

7. Missing Attribute Values: none

8. Class Distribution (number of instances per class)

| class | N | N[%] |
|---|---|---|
| unacc | 1210 | (70.023 %) |
| acc | 384 | (22.222 %) |
| good | 69 | ( 3.993 %) |
| v-good | 65 | ( 3.762 %) |

# Methodology –

## Data Set upload –

The dataset is uploaded to the google colab file.

## Data Pre-Processing –

The Data is first converted to Data Frame and then checked for type of data entries and features.
After that, the data is split into train and test data and then Label Encoding is done for the data entries.

## Model Selection -

After Data Pre-Processing the main objective is to select the Model to predict and train the Data set. Here, we have done classification and chosen Random Forest Classifier, K-Neighbors Classifier and Decision Tree Classifier.

## Data Presentation –

The Data set is fit into these algorithms and then the result is displayed in form of training accuracy and testing accuracy and also by plotting the confusion matrix for all models.

# Results –

**Classification Report :**

```
from sklearn.metrics import classification_report
cr=classification_report(y_test,y_pred_test)
print(cr)
```

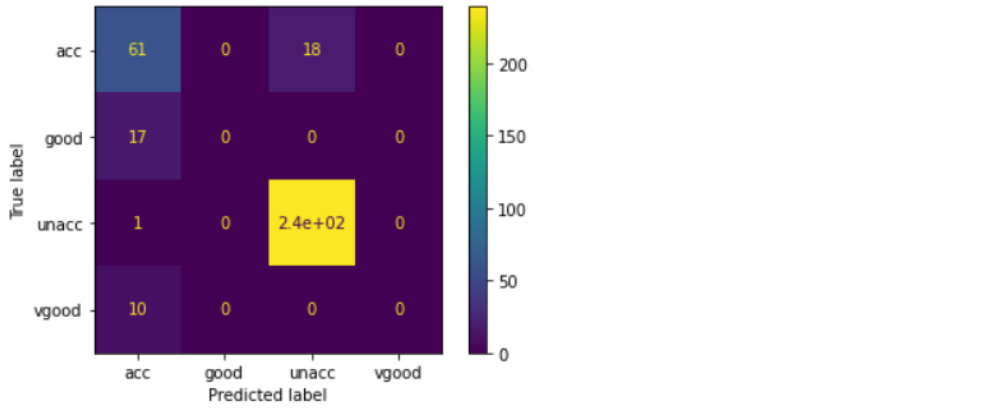
```
              precision    recall  f1-score   support

         acc       0.69      0.77      0.73        79
        good       0.00      0.00      0.00        17
       unacc       0.93      1.00      0.96       240
       vgood       0.00      0.00      0.00        10

    accuracy                           0.87       346
   macro avg       0.40      0.44      0.42       346
weighted avg       0.80      0.87      0.83       346
```

## Confusion Matrix :

**For Random Forest Classifier -**

```
plot_confusion_matrix(rfc,x_test,y_test)
```

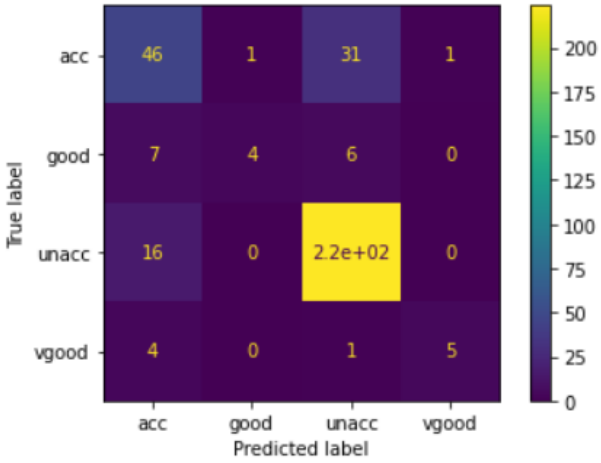
```
[[ 61   0  18   0]
 [ 17   0   0   0]
 [  1   0 239   0]
 [ 10   0   0   0]]
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f52483a1410>
```



## For K-Neighbors Classifier -

```
plot_confusion_matrix(knc,x_test,y_test)
```

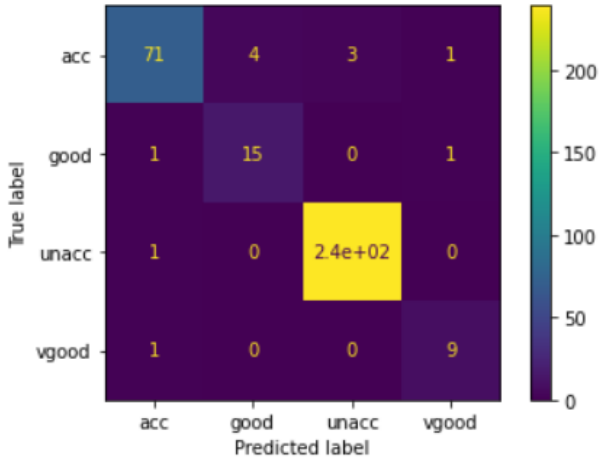
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f5248475490>
```




## For Decision Tree Classifier -

```
plot_confusion_matrix(dtc,x_test,y_test)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f5247d1fb50>
```



# Results :

**For the given Data Set all these Classification Algorithms performed Similarly**

```
from sklearn.metrics import accuracy_score
train_acc=accuracy_score(y_train,y_pred_train)
test_acc=accuracy_score(y_test,y_pred_test)
print('Training Accuracy: ',train_acc)
print('Testing Accuracy: ',test_acc)
```

```
Training Accuracy:  0.8625180897250362
Testing Accuracy:  0.8670520231213873
```

Here we can see that Testing accuracy is not below Training Accuracy so it is not the case of underfitting and overfitting. The accuracies are similar so the model is performing good.

Link for the Colab code File :-

https://colab.research.google.com/drive/1YwsVgxIrIBuBEQ2KWMHwWH29kH5YBycp#scrollTo=r9Kd1Ni0c6tq